REMARKS

In response to the Final Office Action mailed March 14, 2005, Applicant respectfully requests the consideration of the following remarks and, accordingly, the reconsideration of the Final Rejection of the claims. It is respectfully submitted that the pending claims are in condition for allowance.

A 3 month extension of time is also requested, and payment for the fee for a 3 month extension of time is included herein.

Status of the Claims

Claims 1-15 were originally presented for examination. All of the original claims were rejected in the Office Action of July 9, 2004. Applicants previously amended the claims and canceled claims 8-12. This left claims 1-7 and 13-16 which were rejected in the Final Office Action of March 31, 2005.

Applicants have amended the claims to particularly point out and describe their contribution to the art.

The claims now pending are claims 1-7 and 13-16.

Status of Amendments

There are no unentered amendments.

Summary of the Invention

Applicants' claimed invention is a method, system, and program product for controlling access, in real time, to a file among a plurality of users. The first claimed step is establishing an object comprising distinguishable groups of data. Each group of data has associated access criteria. This access criteria controls access to the groups of data. Specifically, each group of data has an associated user privilege for identifying separate groups of information to which the user may have access to within the groups of data and for setting a user's ID. This includes defining which users are allowed to access the object and associated information and user privileges.

The system has a cache memory for storing user ID's; and a cache memory for storing user access criteria along with access application code that is responsive to (1) the user ID, (2) user access criteria associated with the groups of data contained within an object, and (3) predetermined privileges for allowing controlled access to individual groups of data contained within the object by an individual user according to the user's access privileges. These various memories are searched sequentially for the user ID, the access criteria, and the target data.

Thereafter the user is served with a redacted copy of the object, that is, the system sends an electronic object to the user that contains the groups of information to which the user has access and that excludes groups of information to which the user does not have access.

The Rejections

The art rejections are summarized in the table below:

Claim Number	103 – Mukherjee in view of Sadovsky	103 – Mukherjee in view of Kimura	103 – Mukherjee in view of Sadovsky and Koko
1	X		
2	X		
3			X
4			X
5			X
6	X		
7		X	
13		X	
14		X	
15		X	
16	X		

It is conceded in the Final Action that Mukherjee et al does not show cache memory, but that Sadovsky et al. shows a cache memory. It is also conceded that Mukherjee et al. does not teach

> *"identifying a user to have access to the object, verifying a user's user privilege access criteria including extracting the user's user identification from the object request, verifying first in cache memory and if not in cache then in main memory the user's user identification"*

and

> *"searching first in cache and if not found in memory then in main memory"*

but that Kimura et al. overcomes the second deficiency, and that hierarchy of memory is well known in the art.

The References

The primary reference relied upon is United States Patent to Mukherjee et al. for Method For The Storage Of Multi-Versioned Data With Retrieval Based On Searched Query. Mukherjee et al. describes a method for storage and retrieval of both time-oriented versions and view-oriented versions of engineering change information. As described by Mukherjee et al. the engineering change information progresses through a set of status conditions. Access to the data by different user groups is conditioned upon the status of the information.

Version control software logic enables users to create versioned objects by logical key grouping of data elements. Mukherjee's version control logic acts upon logical keys and special versioned attributes of these objects for the proper specification and selection of object instances during creation, update or retrieval processing.

Insert and extract sequence numbers are automatically generated for both historical preservation of previous engineering change information and efficient retrieval of the currently effective designs. Instance level security facilitates the merger of different

versions of the engineering change data having different engineering change status levels so that similar types of data can be contained within a single data base table.

The relevant portions of Mukherjee et al. are reproduced in the footnote below[1].

---

[1] US Patent 5,317,729:

Column 4, lines 12-31:
A block diagram implementation of the components of the preferred embodiment of this invention is illustrated in FIG. 1. As shown in this figure, the version control system 10 includes a version control computer program 20 which operates on computer processor 30. Design engineers and other groups of users at terminal device 18 interact through processor 30 with relational data files stored on direct access (non-volatile) storage device (DASD) 40. Permanently stored on DASD 40 are engineering change notices file 12, master item file 14, engineering change affected item file 16, item related data files 22 and bill of material related data files 24. Version control logic 20 controls the storage and retrieval of versioned engineering change data.

A design engineer using terminal device 18 creates engineering change notices 12 which can be divided broadly into three categories: notices affecting item related data 22, notices affecting bill of material related data 24, and notices affecting both item related data 22 and bill of material related data 24.

Column 4, lines 48-51
The present invention provides version control logic 20 that can process different types of objects that are constructed by any desirable grouping of required data elements.

Column 4, lines 64-67
The version control logic 20 provides means to handle the complexity of processing many types of versioned objects and of supporting many types of version-controlled functions as explained below.

Column 5, line 55 – column 6, line 22:
The use of instance level security to control access to engineering change data at different engineering change affected item status levels is shown in FIG. 3. The security implementation can vary based on user group requirements, but, in general, information about products in a development stage (pre-release status) that is accessible by design engineers should be at a more restrictive security level than information about products that are fully developed (released status) and that is accessible by manufacturing engineers. FIG. 3 indicates that design engineers belong to a user group that has access to data at security level 05 which is the most restrictive in this embodiment. Design engineers also have access to data at the less restrictive security levels 01 through 04. Similarly, manufacturing engineers have access to data at security levels 04 and lower corresponding to engineering change affected items 16 in a status of release. Production planners have access to data at security levels 03 and lower corresponding to engineering change affected items 16 that have been "accepted" by the manufacturing group. The production control group has access to data at security level 02 that has been made effective by manufacturing. Finally, once the affected items in an engineering change are closed, data access at the least restrictive security level 01 is provided to an archival library user group.

In order to implement instance level security, a security level attribute is associated with each object instance. Using database terminology, this is equivalent to requiring that every row in a relational data base table have a security level column. By judiciously combining EC affected item status with instance security levels as explained above, different groups of users, having access to data at different status levels for the EC affected items, are presented with different views of the stored data.

Column 6, lines 32-41:

U.S. Patent 5,689,138 to Sadovsky for Method For Providing Access To Independent Network Resources By Establishing Connection Using An Application Programming Interface Function Call Without Prompting The User For Authentication Data describes a method and system for providing access to independent network resources. At system logon, logon data is stored in memory of a client computer. When a server is accessed, server authentication data is stored in a cache. System logon data and authorization data can be applied to access an independent server resource without requiring user interaction.

The first entry in EC affected item file 16 in FIG. 4 has an EC identifier "ECA", an item identifier "A", and affected item status of pre-release. The user defined view identifier "Eng1" specifies that this view of item A is an engineering view. The system generated design sequence number "8001" establishes a unique relationship between ECA and item A. Item effectivity data includes planned effectivity type "Date" and planned effectivity start date of "3-1-90". The product identifier "P1" specifies that assembly A is used in product P1.

Column 8, line 61 to column 9, line 21:
After promoting the status of item A in ECA from released to accepted at manufacturing plant "Loc1", a location view is created as indicated in FIG. 10. A location view allows local restructuring of a bill of material at a manufacturing plant or production line. In the example, the bill of material for assembly item A is initially copied to create a new "Loc1" view of the data as indicated by the last three entries in bill of material file 68. The security level has been changed from 04 to 03 to permit access to this data by the production planning function.

Other implementations of view versions, such as for rework views, may create a view wherein the differences between the manufacturing bill of material and rework bill of material are recorded within the rework view. The version control logic 20 relates the design sequence number for the rework view to the design sequence for the manufacturing view on which the rework view is based. For this reason, "based on view" and "based on sequence number" attributes are included in the location affected item object in location affected item file 16'. These same attributes are included in the EC affected item objects in EC affected item file 16. This enables one engineering view to be based upon another engineering view in a recursive relationship. Multiple view versions can be stacked in a preferential order so that a match can be searched for a particular selection in one view, and then in other views.

Column 9, lines 21-39:
A number of retrieval options are available to the design or manufacturing engineers at terminal 18 when using version control logic 20 to interact with the relational data base files stored on DASD 40. FIG. 11 represents a screen display that is visible on terminal 18 during interactive processing. The user enters item number, object name, and, as appropriate, EC identifier(s) and effective date(s). The first retrieval option, "applicable at specific EC level" is used primarily to perform update verifications and to select a level of EC related data to modify or delete. It can also be used to review data prior to updates at a particular EC level. The version control logic 20 searches the versioned data files for data records satisfying the search criteria (1) that the inserting EC is the specified EC or an earlier one and (2) either the extracting EC is later than the specified EC, or the EC controlled object is unextracted.

U.S. Patent 5, 864,853 to Kimura et al., for Portable File System Operable Under Various Computer Environments describes a portable file system capable of sharing data among various computer environments, such that a user can freely utilize his own data under various computer environments. The system includes at least one portable data processing device having a memory unit for storing file data and file management data of portable side files; at least one stationary data processing device having a processing unit for executing desired processing by making accesses to the portable side files; and a conversion unit for converting at least one of the file data and the file management data of each requested file among the portable side files for which a file access request is issued by the processing unit into a form suitable for the stationary data processing device. As described by Kimura et al. the processing unit makes an access to each requested file among the portable side files according to converted file data/file management data obtained by the conversion unit. The conversion unit may be provided in the portable data processing device side, or in the stationary data processing device side.

The portions of Kimura et al. relied upon in the Final Action are reproduced in the footnote below.[2]

U.S. Patent 5,434,791 to Koko et al. for Product Structure Management describes an object-oriented method of using a computer to store a model of an imprecise structure of a product. The product's components are modeled as items and item revisions. Each item and item revision has a view, which may have view revisions. The method links

---

[2] U.S. Patent 5,864,853:

Column 13, lines 18-28:
Then, the access request processing unit 121 checks the access right by comparing the user ID in the obtained file management data and the user ID of the process which issued the access request, to judge whether it is a proper access or not (S56), and if not (S56 NO), an improper access notice is issued (S57), whereas otherwise (S56 YES), an access preparation with respect to a read request which is expected to be issued subsequently (such as an initial setting of the file position data as a subsequent reading or writing position data, etc.) is carried out (S58) while the file descriptor is returned (S59).

view objects and view revision objects with occurrence references to each other and to view objects and view revision objects of other components. Context-specific view revisions are modeled as appearances. A user's request for a display of a product is received and used to invoke configuration rules that determine which view revisions) are part of the product. The correct view revisions are assembled with their occurrences and appearances.

The portion of Koko relied upon in the Final Office Action is reproduced in the footnote below[3].

[3] U.S. Patent 5,434,791

Column 28, lines 5-10

14. The method of claim 7, further comprising the step of displaying said bill of materials, and receiving data from a user to add, modify, or delete data attached to said view objects.

Issue

The sole issue presented is that the pending claims are properly allowable to Applicants over Mukherjee et al., either alone or with Sadovsky et al. Kimura et al., and/or Koko et al.

Argument

Summary of The Argument

As conceded in the Office Action, Mukherjee et al. fails to teach critical aspects of Applicants' claimed invention, i.e.,

> *"identifying a user to have access to the object, verifying a user's user privilege access criteria including extracting the user's user identification from the object request, verifying first in cache memory and if not in cache then in main memory the user's user identification"*

Applicants' claimed invention is a method, system, and program product for controlling access, in real time, to a file among a plurality of users. The first claimed step is establishing an object comprising distinguishable groups of data. Each group of data has associated access criteria. This access criteria controls access to the groups of data. Specifically, each group of data has an associated user privilege for identifying separate groups of information to which the user may have access to within the groups of data and for setting a user's ID. This includes defining which users are allowed to access the object and associated information and user privileges.

The system has a cache memory for storing user ID's; and a cache memory for storing user access criteria along with access application code that is responsive to (1) the user ID, (2) user access criteria associated with the groups of data contained within an object, and (3) predetermined privileges for allowing controlled access to individual groups of data contained within the object by an individual user according to the user's access privileges.

Moreover, the method and program product claims positive recite extracting the user's user identification from the object request, verifying (first in cache memory and if not in cache then in main memory) the user's user identification and identifying the groups of data to which the user has access and privileges with respect thereto. This allows controlled access to individual groups of data contained within the object by an individual user according to the user's privileges. Next the claims recite searching for the data first in cache and if not found in cache then in main memory and retrieving the data requested according to the user's access criteria.

The claimed invention further claims transmitting a redacted object to the user, i.e., sending an electronic object to the user that contains the groups of information to which the user has access to and that excludes groups of information to which the user does not have access.

By way of contrast, the primary reference, Mukherjee et al. is directed to a "version control system"[4] characterized as a method for implementing the control of versioned

---

[4] Claim 1 of Mukherjee et al recites:
1. A method for implementing the control of versioned data objects affected by engineering changes to a product design in a computer-based information processing system in which a master item file comprised of items resides on a non-volatile storage device with each of said items representing a part to be used to assemble a product, said method comprising:

creating a first engineering change notice that identifies a first plurality of items in a master item file, said first plurality of items comprised of at least a first item, said first item affected by a first engineering change;

creating a first affected item record for said first item and storing said first affected item record in an affected item file on said non-volatile storage device;

identifying said first affected item record by a first sequence number;

determining a first plurality of versioned data objects that are affected by said first engineering change,

modifying each of said first plurality of versioned data objects, said modification including modifying a plurality of version control attributes, said attributes comprising at least a sequence number field and said modification comprising at least modifying said sequence number field to contain said first sequence number, the result of such modification being a first modified plurality of versioned data objects and

data objects affected by engineering changes to a product design in a computer-based information processing system in which a master item file comprised of items resides on a non-volatile storage device with each of said items representing a part to be used to assemble a product..." using "version control logic" (Column 4, line 48) for "processing many types of versioned objects and of supporting many types of version-controlled functions" (column 4, lines 64-67).

The security system of Mukherjee et al, described at column 5, line 55, to column 6, line 41[5], describes security levels in a single, hierarchal system, and not in a multi-vendor system where the multiple vendors are frequently competitors.

---

storing said first modified plurality of versioned data objects in a plurality of versioned data object data files on said non-volatile storage device;

creating a second engineering change notice that identifies a second plurality of items in said master item file, said second plurality of items comprised of at least a second item, said second item affected by a second engineering change;

creating a second affected item record for said at least one item and storing said second affected item record in said affected item file;

identifying said second affected item record by a second sequence number;

determining a second plurality of versioned data objects that are affected by said second engineering change,

modifying each of said second plurality of versioned data objects, said modification comprised of modifying said plurality of version control attributes, said attributes comprising at least said sequence number field and said modification comprising at least modifying said sequence number field to contain said second sequence number, the result of such modification being a second modified plurality of versioned data objects and storing said second modified plurality of versioned data objects in said plurality of versioned data object data files; and

if said first engineering change satisfies a search query, retrieving said first affected item record and said first modified plurality of versioned data objects by means of said first sequence number; and

if said second engineering change satisfies a search query, retrieving said second affected item record and said second modified plurality of versioned data objects by means of said second sequence number.

[5] Column 5, line 55 – column 6, line 22:
The use of instance level security to control access to engineering change data at different engineering change affected item status levels is shown in FIG. 3. The security implementation can vary based on user group requirements, but, in general, information about products in a development stage (pre-release status) that is accessible by design engineers should be at a more restrictive security level than information about products that are fully developed (released status) and that is accessible by manufacturing engineers. FIG. 3 indicates that design engineers belong to a user group that has access to data at security level 05 which is the most restrictive in this embodiment. Design engineers also have

This is an issue that is not addressed by Mukherjee et al. It is not addressed at the access control level, the data base management system level, the schema level, the metadata level, the object level, or the attribute level.

At the most fundamental level, Mukherjee et al. fails to teach or suggest Applicants' claimed invention, and the deficiencies of Mukherjee et al are not remedied by Sadovsky, Kimura, or Koko. Therefore, the claims as amended are allowable over these references.

Objections Under 35 USC§112

It is respectfully submitted that the amendments to the claims obviate the objections to the claims under 35 USC 112, both first and second paragraphs.

---

access to data at the less restrictive security levels 01 through 04. Similarly, manufacturing engineers have access to data at security levels 04 and lower corresponding to engineering change affected items 16 in a status of release. Production planners have access to data at security levels 03 and lower corresponding to engineering change affected items 16 that have been "accepted" by the manufacturing group. The production control group has access to data at security level 02 that has been made effective by manufacturing. Finally, once the affected items in an engineering change are closed, data access at the least restrictive security level 01 is provided to an archival library user group.

In order to implement instance level security, a security level attribute is associated with each object instance. Using database terminology, this is equivalent to requiring that every row in a relational data base table have a security level column. By judiciously combining EC affected item status with instance security levels as explained above, different groups of users, having access to data at different status levels for the EC affected items, are presented with different views of the stored data.

Column 6, lines 32-41:
The first entry in EC affected item file 16 in FIG. 4 has an EC identifier "ECA", an item identifier "A", and affected item status of pre-release. The user defined view identifier "Engl" specifies that this view of item A is an engineering view. The system generated design sequence number "8001" establishes a unique relationship between ECA and item A. Item effectivity data includes planned effectivity type "Date" and planned effectivity start date of "3-1-90". The product identifier "P1" specifies that assembly A is used in product P1.

Conclusion

It is respectfully submitted that the pending claims define an invention that is patentable to Applicants. Entry thereof is respectfully requested.

Claims 1-7 and 12-16 are pending. Applicants respectfully submit that, in view of the discussion set forth herein, the pending claims are patentable over the prior art.

The Commissioner is hereby authorized to charge any additional fees due or credit any overpayment to Deposit Account No. 50-2421.

If there are any impediments to allowance that the examiner thinks can be overcome with a telephone conference, or if there are any questions regarding this correspondence, please contact the undersigned, Dave Stevens, at (408) 288-7588.

Respectfully Submitted,

Dated: September 13, 2005 By:

David R. Stevens
Reg. No. 38,626

Stevens Law Group
99 North First Street, Suite 201
San Jose, CA 95113
PH (408)288-7588
FX (408)288-7542
Dave.Stevens@StevensLawGroup.com
www.stevenslawgroup.com